# Linux based 3G Multimedia Mobile-phone API Specification [Equipment Service]

## Draft 1.0

NEC Corporation

Panasonic Mobile Communication Ltd.

Contents

# Preface

This document describes an API specification of Equipment service for Linux based 3G multimedia mobile-phone. This document is the results of the work of the CE Linux Forum's technical working group. The APIs in this document are based on the technology which is originally the collaborative work by NEC Corporation, Panasonic Mobile Communication Ltd., and NTT DoCoMo, Inc.

# 1. OBEX

## 1.1 Client send Request

| Message | Client Send Request | |
|---|---|---|
| Message ID | MOE_CLI_MSGID | |
| Operation Classification | Native Client | |
| Functional Overview | | |
| Start the Client send operation.<br>- If the Send Object Classification (Refer to Table 2.4.10.2) is a 1 item object, connect without authentication and if it is all items object, connect using normal authentication.<br>- Send the object designated by the Send Object type.<br>- If it is a 1 item object, send the 1 item object indicated by the "Folder Storage No." and "Memory Storage No.".<br>- Send Client Send Request Confirmation Notice to the Superior APL. | | |
| Message Structure | typedef struct tagMOE_CLI_MSGID{<br>int msgid;<br>int mode;<br>unsigned short object;<br>unsigned short folderno;<br>unsigned short memno;<br>unsigned short session;<br>unsigned char len;<br>char dial[MOE_ISDN_DIAL_MAX];<br>}_MOE_CLI_MSGID; | |
| Parameter | Type | Description |
| Msgid | Int | Message Type<br>Set MOE_CLI_MSGID |
| Mode | Int | Mobile Device Mode<br>MOE_NORMAL_MODE (0): Normal Mode<br>MOE_SECRET_MODE (1): Secret Mode<br>MOE_SECRET_SMODE (2): Secret Exclusive Mode |
| Object | unsigned short | Send Object Type Table 1.3 for reference |
| Folderno | unsigned short | Folder Storage No.<br>Folder Referable No. (Level 1 only) |
| Memno | unsigned short | Memory Storage No.<br>Memory Referable No. (Level 1 only) |
| Session | unsigned short | Session Number<br>Session Number during Authentication (Level 2 only) |
| Len | unsigned char | Local Number Dial Length |

| | | |
|---|---|---|
| Dial | char[MOE_ISDN_DIAL_MAX]; | Local Number Dial<br>MOE_ISDN_DIAL_MAX is the maximum number of digits that can be dialed. |
| Remark | | |
| If the Send Object type is "Tag designated send", the memory storage number and folder storage number should be 0.<br>If the send object is a forbidden operation (locked) or if there is no send object, do not send this message.<br>Only set objects (video) that need a folder storage number. If not necessary, be sure to set it at 0. | | |

## 1.2 Server Request

| Message | Server Request | |
|---|---|---|
| Message ID | MOE_SVR_MSGID | |
| Operation Classification | Native Server | |
| Functional Overview | | |
| Start Server operation.<br>- Start Server operation.<br>- When OBEX forwarding is finished, notify Superior APL of the communication results using "OBEX Communication Result Display Notification". | | |
| Message Structure | typedef struct tagMOE_SVR_MSGID{<br>int msgid;<br>unsigned char result;<br>int mode;<br>unsigned char auth;<br>unsigned short session;<br>unsigned char len;<br>char dial[MOE_ISDN_DIAL_MAX];<br>}_MOE_SVR_MSGID; | |
| Parameter | Type | Description |
| Msgid | Int | Message Type<br>Set MOE_SVR_MSGID |
| Result | unsigned char | Activation Result<br>MOE_START_FAIL (0): Activation Failure<br>MOE_START_SUCCESS (1): Activation Failure |
| Mode | Int | Mobile Device Mode<br>MOE_NORMAL_MODE (0): Normal Mode<br>MOE_SECRET_MODE (1): Secret Mode<br>MOE_SECRET_SMODE (2): Secret Exclusive Mode |
| Auth | unsigned char | Authentication Mode<br>MOE_MODE_AUTH_OFF (0): No Authentication<br>MOE_MODE_AUTH_ON(1):Normal Authentication<br>MOE_MODE_AUTH_SYNC (2): IrMC-Sync Authentication (ALADIN) |
| Session | unsigned short | Session number (session)<br>Only the normal authentication needs a session number. |
| Len | unsigned char | Local Number Dial Length |
| Dial | char[MOE_ISDN_DIAL_MAX]; | Local Number Dial<br>MOE_ISDN_DIAL_MAX is the maximum number of digits that can be dialed. |

| | | |
|---|---|---|
| | | |

Remark

Refer to Table1-1 for details on the parameter designation method.
In the Server condition, OBEX checks for the forbidden operations shown below and judges whether or not OBEX forwarding in possible. Do not conduct check for Superior applications.

- Dial Transmission Forbidden

Also, in the case of the forbidden operations shown below, check the superior applications and send a message of the activation results as activation failure.

- All Lock
- Self Mode
- PIM Lock

Table 1-1 The parameter of the Server Request for AP Activation Request
O : Designation necessary X: Designation not necessary

| Server Request parameter settings | | In the case of memory forwarding applications | | | In the case of Infrared forwarding applications | |
|---|---|---|---|---|---|---|
| | | No Authentication | Normal Authentication | IrMC-Sync Authentication | Normal Reception | All Items Forwarding (Reception) |
| Successful | Activation Result | Activation Successful | | | Activation Successful | |
| | | | | | | |
| | Mobile Device Mode | OX | O | O | O | O |
| | Authentication Mode | No Authentication | Normal Authentication | IrMC-Sync Authentication | No Authentication | Normal Authentication |
| | Session Number | X | O | X | X | O |
| | Local Number | O | O | O | O | O |
| Failure | Activation Result | Activation Failure | | | Activation Failure | |
| | | | | | | |
| | Mobile Device Mode | X | X | X | X | X |
| | Authentication Mode | No Authentication | Normal Authentication | IrMC-Sync Authentication | No Authentication | Normal Authentication |
| | Session Number | X | X | X | X | X |
| | Local Number | X | X | X | X | X |

## 1.3 Abort Request

| Message | Abort Request | |
|---|---|---|
| Message ID | MOE_ABORT_REQ_MSGID | |
| Operation Classification | Native Client Native Server | |
| Functional Overview | | |
| Abort OBEX forwarding.<br>- Abort OBEX forwarding.<br>- Send "Abort Request Notification" message to superior APL. | | |
| Message Structure | int msgid; | |
| Parameter | Type | Description |
| Msgid | int | Message Type<br>Set MOE_ABORT_REQ_MSGID |
| Remark | | |
| Use when OBEX forwarding is aborted by system and UI. | | |

## 1.4 1 Item Registration Request/ All Data Receive Request

| Message | 1 Item Registration Request/All Data Receive Request | |
|---|---|---|
| Message ID | MOE_CONF_MSGID | |
| Operation Classification | Native Server | |
| Functional Overview | | |

1 Item Registration Request:
If a 1 item object is received in the Server condition, use to notify whether or not the received object will be registered in the database in the mobile device. Refer to Sequence 2.6.2, 2.6.6, and 2.6.7.
If the registration confirmation UI is time-out, notify as NO.

[If the Registration Confirmation Information is YES]
- Conduct the registration of the received object or the activation of JavaAppli.
- Notify the registration result using the "OBEX Communication Result Display Notification" message.

[If the Registration Confirmation Information is NO]
- Destroy the received object.

- Notification of the registration result is not conducted because registration is not conducted.

All Data Receive Request:
Just before the reception of all objects starts in the Server condition, use to notify whether or not the received objects will be overwritten in the database in the mobile device. Refer to Sequence 2.6.8 and 2.6.9.
If the registration confirmation UI is time-out, notify as NO.

[If the Registration Confirmation Information is YES]
- Start the reception of the received object.
- After reception is complete, notify the registration result using the "OBEX Communication Result Display Notification" message.

[If the Registration Confirmation Information is NO]
- Destroy the received object.

- OBEX is in the reception ready state (Server).

| Message Structure | typedef struct tagMOE_CONF_MSGID{<br>int msgid;<br>unsigned char proc;<br>}_MOE_CONF_MSGID; | |
|---|---|---|
| Parameter | Type | Description |
| Msgid | int | Message Type<br>Set MOE_CONF_MSGID |

| | | |
|---|---|---|
| Proc | unsigned char | Registration Confirmation Information<br>MOE_NO (0): NO<br>MOE_YES (1): YES |
| Remark | | |
| | | |

## 1.5 AP Start Activation Request

| | | |
|---|---|---|
| Message | AP Activation Start Request | |
| Message ID | MOE_OBEXSTART_MSGID | |
| Operation Classification | Native Server | |
| Functional Overview | | |
| If a CONNECT packet is received from the recipient device, OBEX sends this message to the Superior APL.<br>If the "AP Start Activation Request" message is received, make sure the superior APL conducts the following operations.<br>- Send "Server Request" message to OBEX. | | |
| Message Structure | typedef struct tagMOE_OBEXSTART_MSGID{<br>int msgid;<br>unsigned char auth;<br>} _MOE_OBEXSTART_MSGID; | |
| Parameter | Type | Description |
| Msgid | int | Message Type<br>Set MOE_OBEXSTART_MSGID |
| Auth | unsigned char | Authentication Mode<br>MOE_MODE_AUTH_OFF (0): No Authentication<br>MOE_MODE_AUTH_ON (1): Normal Authentication<br>MOE_MODE_AUTH_SYNC (2): IrMC-Sync Authentication (ALADIN) |
| Remark | | |

## 1.6 Client Send Request Confirmation Notice

| Message | Client Send Request Confirmation Notice | |
|---|---|---|
| Message ID | MOE_CLI_IND_MSGID | |
| Operation Classification | Native Client | |
| Functional Overview | | |
| If OBEX receives a "Client Send Request" message, send a "Client Send Request Confirmation Notice" message to the superior APL. | | |
| Message Structure | typedef struct tagMOE_CLI_IND_MSGID {<br>  int msgid;<br>  unsigned char result;<br>}_MOE_CLI_IND_MSGID; | |
| Parameter | Type | Description |
| Msgid | int | Message Type<br>Set MOE_CLI_IND_MSGID |
| Result | unsigned char | Results<br>MOE_CLIENT_OK(1): Request Reception |
| Remark | | |
| The "Result" parameter is always "Request Reception".<br>If the connection fails, the "OBEX Communication Result Display Notification" message is sent. | | |

## 1.7 Server Request Confirmation Notice

| Message | Server Request Confirmation Notice | |
|---|---|---|
| Message ID | MOE_SVR_IND_MSGID | |
| Operation Classification | Native Server | |
| Functional Overview | | |
| If OBEX receives a "Server Request" message, send a "Server Request Confirmation Notice" message to the superior APL. | | |
| Message Structure | typedef struct tagMOE_SRV_IND_MSGID{<br>int msgid;<br>unsigned char result;<br>}_MOE_SRV_IND_MSGID; | |
| Parameter | Type | Description |
| Msgid | int | Message Type<br>Set MOE_SVR_IND_MSGID |
| Result | unsigned char | Results<br>MOE_SERVER_OK(1): Request Reception |
| Remark | | |
| This message notifies superior APL only when connection is successful, therefore the "Result" parameter is always "Request Reception". If the connection fails, the "OBEX Communication Result Display Notification" message is sent. | | |

## 1.8 Abort Request Notification

| | | |
|---|---|---|
| Message | Abort Request Notification | |
| Message ID | MOE_ABORT_IND_MSGID | |
| Operation Classification | Native Server, Native Client | |
| Functional Overview | | |
| If OBEX receives a "Abort Request" message, abort OBEX forwarding and send "Abort Request Notification" to superior APL. | | |
| Message Structure | int msgid; | |
| Parameter | Type | Description |
| Msgid | int | Message Type<br>Set MOE_ABORT_IND_MSGID |
| Remark | | |

## 1.9 OBEX Forwarding Status Display Notification

| | | |
|---|---|---|
| Message | OBEX Forwarding Status Display Notification | |
| Message ID | MOE_PROGRESS_MSGID | |
| Operation Classification | Native Server, Native Client, JavaAppli Communication Server, JavaAppli Communication Client | |
| Functional Overview | | |
| Notify the progress during PUT or GET. | | |
| Message Structure | typedef struct tag MOE_PROGRESS_MSGID {<br>int msgid;<br>unsigned short object;<br>unsigned char TxRx;<br>signed short count;<br>}_MOE_PROGRESS_MSGID; | |
| Parameter | Type | Description |
| Msgid | int | Message Type<br>Set MOE_PROGRESS_MSGID |
| Object | unsigned short | Object type (Native) Refer to Table 2.4.10.2<br>Object type (JavaAppli Communication) Notify MOE_TYP_NONE |
| TxRx | unsigned char | Send and Receive (Native)<br>MOE_MODE_OBJSEND (0):send --- ServerGET, ClientPUT<br>MOE_MODE_OBJRECV (1):receive --- ServerPUT<br>- Send and receive (JavaApplication communication)<br>MOE_MODE_OBJSEND (0): send --- ServerGET, ClientPUT<br>MOE_MODE_OBJRECV (1): receive --- ServerPUT, Client GET |
| Count | signed short | Progress %<br>0～100<br>-1 does not conduct forwarding status notification<br>After Infrared LEV2PUT reception, notify 0 during delete completion |
| Remark | | |
| Use for both native application and JavaAppli communication.<br>Always conduct notification during send and reception of the first packet.<br>If the progress does not change after sending and receiving OBEX packet, do not notify.<br>If the header necessary to calculate the progress during OBEX packet was not included, the notification of the progress in not conducted. | | |

The notification timing is as follows.
Server: During the reception of the PUT command and when sending the GET response
Client: When sending the PUT command and when receiving the GET response

## 1.10 OBEX Communication Result Display Notification

| | | |
|---|---|---|
| Message | OBEX Communication Result Display Notification | |
| Message ID | MOE_RESULT_MSGID | |
| Operation Classification | Native Server Native Client | |
| Functional Overview | | |
| When the communication ends, notify the communication results, object classification and object name.<br>After communication ends, notify that the UI of the registration of the 1 item object is necessary.<br>When Infrared communication starts, notify that the UI of the overwrite confirmation of the all item object is necessary. (All Data Reception Notification) | | |
| Message Structure | typedef struct tagMOE_RESULT_MSGID{<br>int msgid;<br>unsigned short status;<br>unsigned short object;<br>unsigned char name[MOE_OBJNAME_SIZE];<br>unsigned char TxRx;<br>}_MOE_RESULT_MSGID; | |
| Parameter | Type | Description |
| Msgid | int | Message Type<br>Set MOE_RESULT_MSGID |
| Status | unsigned short | Communication Result Refer to Table 1-2. |
| Object | unsigned short | Object type Refer to Table 1-3. |
| Name | unsigned char [MOE_OBJNAME_SIZE] | Object name (19 bytes) Refer to table 1-4.<br>NULL stop available<br>MOE_OBJNAME_SIZE= 19 |
| TxRx | unsigned char | Send and Receive<br>MOE_MODE_OBJSEND (0): send --- ServerGET, ClientPUT<br>MOE_MODE_OBJRECV (1): receive --- ServerPUT<br>MOE_MODE_OBJNONE (2): Not designated --- Server not received |
| Remark | | |
| If more than one object is sent or received, notify the information for the last object that was transferred.<br>Receive the "Server Request" message and the "Client Send Request" message and conduct notification using this message even if the connection fails.<br>Conduct notification using this message even when notifying disconnection error before starting communication (such as before sending the "AP Activation Start Request" message),<br>Example: If the PUT packet is received without receiving the CONNECT packet, send this message and do not send the "AP Activation Start Request" message.<br>When a 1 Item object is received, also send notification that registration confirmation is necessary for the "Communication Result" parameter. | | |

The "Object Name" parameter receives1 Item Object and if the "Communication Result" is "Registration Confirmation Request", notification is conducted.

<When a All Item Object is received during Infrared Server connection>
The "Communication Result" parameter should be "All Data Reception Notification".
Refer to Sequence 2.5.8, 2.5.9, 2.5.11, 2.5.11

Table 1-2 Communication Result Contents (OBEX Communication Result Display Notification)

| Communication Result | Code | Symbol unsigned short Type | Target object | Server PUT | Server GET | Client PUT |
|---|---|---|---|---|---|---|
| Normal Completion | 0x0000 | MOE_CHGCODE_OK | ALL | O | O | O |
| Normal Completion (Reset after forwarding) | 0x1000 | MOE_CHGCODE_OK_RST | Internal Line Setting Utility | O | X | X |
| Authentication Error | 0x0001 | MOE_CHGCODE_UNAUTH | Including ALL and Error objects | O | O | O |
| DatabaseFull error (MAX number of items, area shortage) | 0x0002 | MOE_CHGCODE_DBFULL | ALL | O | X | O |
| DatabaseLocked error | 0x0003 | MOE_CHGCODE_DBLOCK | Including ALL and Error objects | O | O | O |
| Reservation (PIM lock) | 0x0005 | MOE_CHGCODE_PIMLOCK | ALL | O | O | X |
| Transmission Forbidden Error | 0x0006 | MOE_CHGCODE_HASSIN | Address Book | O | O | X |
| Designated Transmission Restriction Error | 0x0007 | MOE_CHGCODE_SITEI | Address Book | O | X | X |
| Multiple Errors | 0x0008 | MOE_CHGCODE_DOUBLE | Schedule/Bookmark | O | X | X |
| Time Out "You have aborted the application. Would you like to continue the abort process?" | 0x0009 | MOE_CHGCODE_TIMEOUT | Including ALL and Error objects | O | O | O |
| Reservation(NotFound) | 0x000A | | | | | |
| Reservation(BadRequest) | 0x000B | | | | | |
| Mobile Device Date and Hour Not Set error | 0x000C | MOE_CHGCODE_NOTSET | Schedule/ToDo | O | X | X |

| | | | | | | |
|---|---|---|---|---|---|---|
| File Attachment Not Possible | 0x000D | MOE_CHGCODE_TEMPERR | Mail, Address Book | O | X | X |
| Registration Not Possible error (Grammar error, Buffer Overflow) | 0x000E | MOE_CHGCODE_TOUROKUERR | Including ALL and Error objects | O | X | X |
| (All Data Reception Notification) | 0x0400 | MOE_CHGCODE_RECEIVE_OK | ALL (All items) | O | X | X |
| Disconnect error (Transmission path interruption) "You have aborted the application. Would you like to continue the abort process?" | 0x00FE | MOE_CHGCODE_ABORT | Including ALL and Error objects | O | O | O |
| Unauthorized Reception Error (Unauthorized command, response) "Transmission not possible." | 0x00FF | MOE_CHGCODE_NG | Including ALL and Error objects | O | O | O |
| | | | | | | |

Recipient Detection error is detected on the APL side.

If it has been disconnected by user operation, the "Abort Request Notification" message is sent.

If time out occurs, be sure to conduct USB or infrared disconnection from the superior APL.

If the file attachment not possible is Server PUT, notification is sent after OBEX receives a 1 item Registration Request (YES) and conducts the 1 item Registration process.

ALL means that all normal objects are subject to the process. Also, for details concerning the error object, refer to Table 1-2.

For the proper wording of the communication results, refer to the separate Wording Specifications.

Table 1-3 Object Type Contents (Client Send Request, OBEX Communication Result Display Notification, OBEX Forwarding Status Display Notification)

| | Object Type | Code | Symbolunsigned shortType | Client Send Request | OBEX Communication Result Display Notification | | | OBEX Forwarding Status Display Notification | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Client PUT | Server PUT | Server GET | Client PUT | Server PUT | Server GET |
| 1 item Object (Level 1) | Address Book | 0x0010 | MOE_TYP_PB | O | O | O | X | O | O | - |
| | Personal Data (Normal) | 0x0040 | MOE_TYP_PR1 | O | O | - | - | O | - | - |
| | Personal Data (Display All) | 0x0041 | MOE_TYP_PRA | O | O | - | - | O | - | - |
| | Schedule/ToDo (vCallender) | 0x0020 | MOE_TYP_CAL | - | - | X | X | - | O | - |
| | Schedule | 0x0021 | MOE_TYP_SCA | O | O | O | - | O | - | - |
| | ToDo | 0x0022 | MOE_TYP_TO | O | O | O | - | O | - | - |
| | Received Mail | 0x0031 | MOE_TYP_IM | O | O | O | X | O | O *1 | - |
| | Saved Mail | 0x0032 | MOE_TYP_OM | O | O | O | X | O | - | - |
| | Send Mail | 0x0033 | MOE_TYP_SM | O | O | O | X | O | - | - |
| | Text Memo | 0x0060 | MOE_TYP_NT | O | O | O | X | O | O | - |
| | Melody | 0x0071 | MOE_TYP_MLD | O | O | O | X | O | O | - |
| | Still Image | 0x0072 | MOE_TYP_IMG | O | O | O | X | O | O | - |
| | Video | 0x0073 | MOE_TYP_VCP | O | O | O | X | O | O | - |
| | Tag Designated Send | 0x00e0 | MOE_TYP_TAG | O | O | - | - | O | - | - |
| | Address Book | 0x1010 | MOE_TYP_PBALL | O | O | O | O | O | O | O |
| | Schedule/ToDo (vCallender) | 0x1020 | MOE_TYP_CALALL | - | - | O | O | O | O | O |
| | Schedule | 0x1021 | MOE_TYP_SCAALL | O | O | X*2 | O | O | - | O |
| All item Object (Level 2) | ToDo | 0x1022 | MOE_TYP_TOALL | O | O | X*2 | O | O | - | O |
| | Received Mail | 0x1031 | MOE_TYP_IMALL | O | O | O | O | O | O | O |
| | Saved Mail | 0x1032 | MOE_TYP_OMALL | O | O | O | O | O | O | O |
| | Send Mail | 0x1033 | MOE_TYP_SMALL | O | O | O | O | O | O | O |
| | Text Memo | 0x1060 | MOE_TYP_NTALL | O | O | O | O | O | O | O |
| | Melody | 0x1071 | MOE_TYP_MLDALL | - | - | O | O | - | O | O |
| | Still Image | 0x1072 | MOE_TYP_IMG | - | - | O | O | - | O | O |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | ALL | | | | | | | |
| | Video | 0x1073 | MOE_TYP_VCP ALL | - | - | O | O | - | O | O |
| | Bookmark | 0x1090 | MOE_TYP_BMA LL | - | - | O | O | - | O | O |
| | Bookmark | 0x1091 | MOE_TYP_IBM ALL | O | O | X*2 | O | O | - | O |
| | Address Book Information Log | 0x2010 | MOE_TYP_PBI NFO | - | - | X | O | - | - | O |
| | Schedule Information Log | 0x2021 | MOE_TYP_SCA INFO | - | - | X | O | - | - | O |
| | ToDo Information Log | 0x2022 | MOE_TYP_TOI NFO | - | - | X | O | - | - | O |
| Inform ation Log Object | Receive Mail Information Log | 0x2031 | MOE_TYP_IMI NFO | - | - | X | O | - | - | O |
| | Saved Mail Information Log | 0x2032 | MOE_TYP_OMI NFO | - | - | X | O | - | - | O |
| | Send Mail Information Log | 0x2033 | MOE_TYP_SMI NFO | - | - | X | O | - | - | O |
| | Text Memo Information Log | 0x2060 | MOE_TYP_NTI NFO | - | - | X | O | - | - | O |
| | Melody Information Log | 0x2071 | MOE_TYP_MLD INFO | - | - | X | O | - | - | O |
| | Still Image Information Log | 0x2072 | MOE_TYP_IMG INFO | - | - | X | O | - | - | O |
| | Video Information Log | 0x2073 | MOE_TYP_VCP INFO | - | - | X | O | - | - | O |
| | Bookmark Information Log | 0x2090 | MOE_TYP_BMI NFO | - | - | X | O | - | - | O |
| | Device Information Log | 0x20a0 | MOE_TYP_DEV INFO | - | - | X | O | - | - | O |
| | Non IrMC file (Error Object) | 0x0000 | MOE_TYP_NON E | - | - | X | X | - | - | - |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | |

The meaning of the symbols is as follows.

O Code that can be designated as a normal object or used in notification

In the case of OBEX Communication Result Display Notification, the communication results MOE_CHGCODE_OK, MOE_CHGCODE_CONFOK, MOE_CHGCODE_RECEIVE_OK are
normal objects.

All OBEX Forwarding Status Display Notifications are normal objects.

X Codes that have a possibility of being notified as an error object

*If the communication results of the 2 symbols are MOE_CHGCODE_DBFULL, distinguish the objects that are DatabaseFull.

- Codes that cannot be designated or used in notification

Note: In OBEX Forwarding Status Display Notification, display "Mail" on the screen during OBEC Communication as the object type for objects with the *1 symbol.

Normal Object

| Object Level -> | 1 item (LEV1) | | All items (LEV2) | | Information Log (LEV2 rewrite not possible) | |
|---|---|---|---|---|---|---|
| Weight | 0x0000 | MOE_TYP_LV1 | 0x1000 | MOE_TYP_LV2 | 0x2000 | MOE_TYP_INFO |
| | | | | | | |

Error Object (OBEX Communication Result Display Notification, notification is sent in the case of Server)

| Object Level -> | LEV3, LEV4 | |
|---|---|---|
| Weight | 0x3000 | MOE_TYP_LV3 |

Table 1-4 Content of the Object name (OBEX Communication Result Display Notification)

| Object | Content of the Object name |
|---|---|
| Address Book 1 item | "Name" (18 bytes) |
| Schedule 1 item | "Content" (18 bytes) |
| ToDo 1 item | "Content" (18 bytes) |
| Mail 1 item | "Subject"(18 bytes) |
| Text Memo1 item | "Text" (18 bytes) |
| Melody 1 item | "Title" (18 bytes) |
| Still Image 1 item | "Title" (18 bytes) |
| Bookmark 1 item | "Title" (18 bytes) |
| Video 1 item | "Title" (18 bytes) |

Note: Notify the contents above only if the communication result is Registration Confirmation Request (0x0300).
The end should be NULL stop.
The area of the object name is 19 bytes.

## 1.11 Client Open Request

| Message | Client Open Request | |
|---|---|---|
| Message ID | MOE_C_OPN_MSGID | |
| Operation Classification | JavaAppli Communication Client | |
| Functional Overview | | |
| Conduct Client Open process of JavaAppli communication.<br>- Send CONNECT packet and connect with receiving device.<br>- Send notification of open result as a "Process Result Notification" message to the superior APL. | | |
| Message Structure | typedef struct tagMOE_C_OPN_MSGID{<br>int msgid;<br>_MOE_IAPLI_BUF *bufadr;<br>}_MOE_C_OPN_MSGID; | |
| Parameter | Type | Description |
| Msgid | Int | Message Type<br>Set MOE_C_OPN_MSGID |
| Bufadr | _MOE_IAPLI_BUF * | Pointer for the Communication Buffer Address<br>Refer to Table 2-5 for the Communication Buffer Address structure. |
| Remark | | |
| A message that corresponds to the Client Request Primitive. | | |

## 1.12 Client PUT Request

| Message | Client PUT Request | |
|---|---|---|
| Message ID | MOE_C_PUT_RQ_MSGID | |
| Operation Classification | JavaAppli Communication Client | |
| Functional Overview | | |
| Conduct PUT process of JavaAppli communication.<br>- Build the header given in the parameter within the OBEX packet and conduct PUT.<br>- When PUT ends (the last PUT packet is sent), send notification to the superior APL using the "Client PUT Response Notification" message.<br>- If a unauthorized response is receive during PUT, send a DISCONNECT packet to the recipient device and notify the Superior APL using the "Process Result Notification" message. | | |
| Message Structure | typedef struct tagMOE_C_PUT_RQ_MSGID {<br>int msgid;<br>signed short namesiz;<br>signed short typesiz;<br>signed short timesiz;<br>unsigned long datasiz;<br>}_MOE_C_PUT_RQ_MSGID; | |
| Parameter | Type | Description |
| msgid | Int | Message Type<br>Set MOE_C_PUT_RQ_MSGID |
| namesiz | signed short | Name header length (-1 to 126) of OBEX |
| typesiz | signed short | Type header length (-1 to 63) of OBEX |
| timesiz | signed short | Time header length (-1, 15, 16) of OBEX |
| datasiz | unsigned long | Send data length (datasiz) (0 to 102400) |
| Remark | | |
| Refer to Table 2-6 for setting values for each header.<br>A message that corresponds to the PUT Request Primitive. | | |

## 1.13 Client GET Request

| | | |
|---|---|---|
| Message | Client GET Request | |
| Message ID | MOE_C_GET_RQ_MSGID | |
| Operation Classification | JavaAppli Communication Client | |
| Functional Overview | | |
| Conduct GET process of JavaAppli communication.<br>- Build the header given in the parameter within the OBEX packet and conduct GET.<br>- If GET achieves normal completion or receives an error response, send notification to the superior APL using the "Client GET Response Notification" message.<br>- If the received data exceeds the regulated size, send a DISCONNECT packet to the recipient device and using the "DISCONNECT Notification", notify the Superior APL that an overflow has occured. | | |
| Message Structure | typedef struct tagMOE_C_GET_RQ_MSGID{<br>int msgid;<br>signed short namesiz;<br>signed short typesiz;<br>signed short timesiz;<br>}_MOE_C_GET_RQ_MSGID; | |
| Parameter | Type | Description |
| Msgid | int | Message Type<br>Set MOE_C_GET_RQ_MSGID |
| Namesiz | signed short | Name header length (-1 to 126) of OBEX |
| Typesiz | signed short | Type header length (-1 to 63) of OBEX |
| Timesiz | signed short | Time header length (-1, 15, 16) of OBEX |
| Remark | | |
| Refer to Table 2.3.24.2 for setting values for each header.<br>A message that corresponds to the GET Request Primitive. | | |

## 1.14 Client Close Request

| Message | Client Close Request | |
|---|---|---|
| Message ID | MOE_C_CLS_MSGID | |
| Operation Classification | JavaAppli Communication Client | |
| Functional Overview | | |
| Conduct Client close process.<br>- Send the DISCONNECT packet to the receiving device.<br>- If a response to the DISCONNECT packet is received from the recipient device, send the "DISCONNECT Notification" message to the Superior APL and notify that disconnection has been completed. | | |
| Message Structure | int msgid; | |
| Parameter | Type | Description |
| Msgid | int | Message Type<br>Set MOE_C_CLS_MSGID |
| Remark | | |
| A message that corresponds with Interruption Primitive<br>Use if it is already opened in the Client. | | |

## 1.15 Server Open Request

| Message | Server Open Request | |
|---|---|---|
| Message ID | MOE_S_OPN_MSGID | |
| Operation Classification | JavaAppli Communication Server | |
| Functional Overview | | |
| Conduct Server Open Request.<br>- It becomes the Server condition.<br>- Send notification of open result as a "Process Result Notification" message to the superior APL. | | |
| Message Structure | typedef struct tagMOE_S_OPN_MSGID{<br>int msgid;<br>_MOE_IAPLI_BUF *bufadr;<br>}_MOE_S_OPN_MSGID; | |
| Parameter | Type | Description |
| Msgid | int | Message Type<br>Set MOE_S_OPN_MSGID |
| Bufadr | _MOE_IAPLI_BUF * | Pointer for the Communication Buffer Address<br>Refer to Table 2.4.24.1 for the Communication Buffer Address structure. |
| Remark | | |
| A message that corresponds with the Server Request Primitive | | |

## 1.16 Server PUT Response Request

| | | |
|---|---|---|
| Message | Server PUT Response Request . | |
| Message ID | MOE_S_PUTRESP_RQ_MSGID | |
| Operation Classification | JavaAppli Communication Server | |
| Functional Overview | | |
| Send PUT response and complete PUT process.<br>- Build the header designated in the parameter in the OBEX packet and send it to the recipient device as a PUT response.<br>- In the "Process Result Notification" message, notify the Superior APL that the PUT process is complete. | | |
| Message Structure | typedef struct tagMOE_S_PUTRESP_RQ_MSGID{<br>int msgid;<br>unsigned char respcode;<br>signed short namesiz;<br>signed short typesiz;<br>signed short timesiz;<br>}_MOE_S_PUTRESP_RQ_MSGID; | |
| Parameter | Type | Description |
| Msgid | int | Message Type<br>Set MOE_S_PUTRESP_RQ_MSGID |
| Respcode | unsigned char | Response Code refer to Table 2.4.24.3 |
| Namesiz | signed short | Name header length (-1 to 126) of OBEX |
| Typesiz | signed short | Type header length (-1 to 63) of OBEX |
| Timesiz | signed short | Time header length (-1, 15, 16) of OBEX |
| Remark | | |
| Refer to Table 2-7 for setting values for each header.<br>The OBEX waits to receive this message after sending the "Server PUT ind Notification" message to the Superior APL.<br>A message that corresponds to the PUT Response Primitive. | | |

## 1.17 Server GET Response Request

| | | |
|---|---|---|
| Message | Server GET Response Request | |
| Message ID | MOE_S_GETRESP_RQ_MSGID | |
| Operation Classification | JavaAppli Communication Server | |
| Functional Overview | | |
| Send GET response and complete GET process.<br>- Build the header designated in the parameter in the OBEX packet and send it to the recipient device as a GET response.<br>- In the "Process Result Notification" message, notify the Superior APL that the GET process is complete. | | |
| Message Structure | typedef struct tagMOE_S_GETRESP_RQ_MSGID {<br>int msgid;<br>unsigned char respcode;<br>signed short namesiz;<br>signed short typesiz;<br>signed short timesiz;<br>unsuigned long datasiz;<br>}_MOE_S_GETRESP_RQ_MSGID; | |
| Parameter | Type | Description |
| Msgid | int | Message Type<br>Set MOE_S_GETRESP_RQ_MSGID |
| Respcode | unsigned char | Response Code refer to Table 2.4.24.3 |
| Namesiz | signed short | Name header length (-1 to 126) of OBEX |
| Typesiz | signed short | Type header length (-1 to 63) of OBEX |
| Timesiz | signed short | Time header length (-1, 15, 16) of OBEX |
| Datasiz | unsigned long | Send data length (datasiz) (0 to 102400) |
| Remark | | |
| Refer to Table 2.3.24.2 for setting values for each header.<br>The OBEX waits to receive this message after sending the "Server GET ind Notification" message to the Superior APL.<br>A message that corresponds to the GET Response Primitive. | | |

## 1.18 Forced Close Request

<table>
<tr><td>Message</td><td colspan="2">Forced Close Request</td></tr>
<tr><td>Message ID</td><td colspan="2">MOE_CS_CLS_MSGID</td></tr>
<tr><td>Operation Classification</td><td colspan="2">JavaAppli CommunicationServer, JavaAppli Communication Client</td></tr>
<tr><td>Functional Overview</td><td colspan="2"></td></tr>
<tr><td colspan="3">Force abort the JavaAppli communication (Close from a separate thread or user abort)<br>- Abort OBEX forwarding<br>- Send "DISCONNECT Notification" message to Superior APL.</td></tr>
<tr><td>Message Structure</td><td colspan="2">typedef struct tagMOE_CS_CLS_MSGID {<br>int msgid;<br>unsigned char reason;<br>}_MOE_CS_CLS_MSGID;</td></tr>
<tr><td>Parameter</td><td>Type</td><td>Description</td></tr>
<tr><td>Msgid</td><td>int</td><td>Message Type<br>Set MOE_CS_CLS_MSGID</td></tr>
<tr><td>reason;</td><td>unsigned char</td><td>Reason<br>MOE_CLS_INTERRUPTION (0): interruption --- abort by the close method<br>MOE_CLS_USERABORT (1) :USER ABORT --- aborted by the user</td></tr>
<tr><td>Remark</td><td colspan="2"></td></tr>
<tr><td colspan="3">A message that corresponds with Interruption Primitive and "User Abort" detection.<br>Use when it is already opened by the Client or Server and you want to conduct forced termination.</td></tr>
</table>

## 1.19 Client PUT Response Notification

| | | |
|---|---|---|
| Message | Client PUT Response Notification | |
| Message ID | MOE_C_PUTRESP_IND_MSGID | |
| Operation Classification | JavaAppli Communication Client | |
| Functional Overview | | |
| Notify the response to the last PUT packet sent during the Client PUT process.<br>- If the response packet for the last PUT packet is received, notify the last response code received.<br>- Notify the header received during the PUT process. | | |
| Message Structure | typedef struct tagMOE_C_PUTRESP_IND_MSGID{<br>int msgid;<br>unsigned char respcode;<br>signed short namesiz;<br>signed short typesiz;<br>signed short timesiz;<br>}_MOE_C_PUTRESP_IND_MSGID; | |
| Parameter | Type | Description |
| Msgid | int | Message Type<br>Set MOE_C_PUTRESP_IND_MSGID |
| respcode | unsigned char | Response code refer to Table 2.4.24.3 |
| Namesiz | signed short | Name header length (-1 to 126) of OBEX |
| Typesiz | signed short | Type header length (-1 to 63) of OBEX |
| Timesiz | signed short | Time header length (-1, 15, 16) of OBEX |
| Remark | | |
| Refer to Table 2.4.24.2 for setting values for each header.<br>If the same header is received multiple times, notify the first header received.<br>A message that corresponds with ResponseData primitive. | | |

## 1.20 Client GET Response Notification

| | | |
|---|---|---|
| Message | Client GET Response Notification . | |
| Message ID | MOE_C_GETRESP_IND_MSGID | |
| Operation Classification | JavaAppli Communication Client | |
| Functional Overview | | |
| Notify the response to the last GET packet sent during the Client GET process.<br>- If the response packet for the last GET packet is received, notify the last response code received.<br>- Notify the header received during the GET process. | | |
| Message Structure | typedef struct tagMOE_C_GETRESP_IND_MSGID{<br>int msgid;<br>unsigned char respcode;<br>signed short namesiz;<br>signed short typesiz;<br>signed short timesiz;<br>unsuigned long datasiz;<br>}_MOE_C_GETRESP_IND_MSGID; | |
| Parameter | Type | Description |
| Msgid | int | Message Type<br>Set MOE_C_GETRESP_IND_MSGID |
| respcode | unsigned char | Response Code refer to Table 2.4.24.3 |
| Namesiz | signed short | Name header length (-1 to 126) of OBEX |
| Typesiz | signed short | Type header length (-1 to 63) of OBEX |
| Timesiz | signed short | Time header length (-1, 15, 16) of OBEX |
| datasiz; | unsigned long | Receive data length (datasiz) (0 to 102400) |
| Remark | | |
| Refer to Table 2.4.24.2 for setting values for each header.<br>If the same header is received multiple times, notify the first header received.<br>A message that corresponds with ResponseData primitive. | | |

## 1.21 Server PUT ind (indication) Notification

| | | |
|---|---|---|
| Message | Server PUT ind (indication) notification | |
| Message ID | MOE_S_PUT_IND_MSGID | |
| Operation Classification | JavaAppli Communication Server | |
| Functional Overview | | |
| Notify the Superior APL of PUT packet reception in the Server state.<br>- Analyze the header from the PUT packet and notify the Superior APL.<br>- Wait for the "ServerPUT response request" message. | | |
| Message Structure | typedef struct tagMOE_S_PUT_IND_MSGID{<br>int msgid;<br>signed short namesiz;<br>signed short typesiz;<br>signed short timesiz;<br>unsigned long datasiz;<br>}_MOE_S_PUT_IND_MSGID; | |
| Parameter | Type | Description |
| Msgid | int | Message Type<br>Set MOE_S_PUT_IND_MSGID |
| Namesiz | signed short | Name header length (-1 to 126) of OBEX |
| Typesiz | signed short | Type header length (-1 to 63) of OBEX |
| Timesiz | signed short | Time header length (-1, 15, 16) of OBEX |
| datasiz; | unsigned long | Receive data length (datasiz) (0 to 102400) |
| Remark | | |
| Refer to Table 2.3.24.2 for setting values for each header.<br>If the "Server PUT ind Notification" message is received, the Superior APL requests the sending of the PUT response to OBEX using the "ServerPUT response request" message.<br>A message that corresponds with RequestData primitive. | | |

## 1.22 Server GET ind (indication) Notification

| | | |
|---|---|---|
| Message | Server GET ind (indication) notification | |
| Message ID | MOE_S_GET_IND_MSGID | |
| Operation Classification | JavaAppli Communication Server | |
| Functional Overview | | |
| Notify the Superior APL of GET packet reception in the Server state.<br>- Analyze the header from the GET packet and notify the Superior APL.<br>- Wait for the "ServerPUT response request" message. | | |
| Message Structure | typedef struct tagMOE_S_GET_IND_MSGID{<br>int msgid;<br>signed short namesiz;<br>signed short typesiz;<br>signed short timesiz;<br>}_MOE_S_GET_IND_MSGID; | |
| Parameter | Type | Description |
| Msgid | int | Message Type<br>Set MOE_S_GET_IND_MSGID |
| Namesiz | signed short | Name header length (-1 to 126) of OBEX |
| Typesiz | signed short | Type header length (-1 to 63) of OBEX |
| Timesiz | signed short | Time header length (-1, 15, 16) of OBEX |
| Remark | | |
| Refer to Table 2.4.24.2 for setting values for each header.<br>If the "Server GET ind Notification" message is received, the Superior APL requests the sending of the GET response to OBEX using the "ServerGET response request" message.<br>A message that corresponds with RequestData primitive. | | |

## 1.23 Process Result Notification

| Message | Process Result Notification | |
|---|---|---|
| Message ID | MOE_RESULT_IND_MSGID | |
| Operation Classification | JavaAppli CommunicationServer, JavaAppli Communication Client | |
| Functional Overview | | |
| Notify the Superior APL of the process results for the following messages.<br>- Server Open Request<br>- Client Open Request<br>- Server PUT Response Request<br>- Server GET Response Request<br>- Client PUT Request<br>- ClientGET Request<br>- Client Close Request | | |
| Message Structure | typedef struct tagMOE_RESULT_IND_MSGID{<br>int msgid;<br>unsigned char result;<br>}_MOE_RESULT_IND_MSGID; | |
| Parameter | Type | Description |
| msgid | int | Message Type<br>Set MOE_RESULT_IND_MSGID |
| result | unsigned char | Communication Result<br>MOE_RESCODE_OK (0) : Normal --- Normal (Request Received)<br>MOE_RESCODE_INCOMPLETE (1) :Incomplete --- Abnormal (Request Failed)<br>MOE_RESCODE_BUSY (2) :Busy --- During DISCONNECT process |
| Remark | | |
| If the "Communication Result" parameter for this message is not "Normal", wait for the "DISCONNECT Notification" message and then activate the call control or retry the infrared communications or display the confirmation screen.<br>A message that corresponds with Accepted, Response Incomplete, Response Complete, Request Incomplete, and Busy primitive. | | |

## 1.24 DISCONNECT Notification

| Message | DISCONNECT Notification | |
|---|---|---|
| Message ID | MOE_DISCONNECT_MSGID | |
| Operation Classification | JavaAppli Communication Server, i33Appli Communication Client, Native Server, Native Client, | |
| Functional Overview | | |
| Notify the Superior APL that JavaAppli communication is complete. | | |
| Message Structure | typedef struct tagMOE_DISCONNECT_MSGID{<br>int msgid;<br>unsigned char result;<br>}_MOE_DISCONNECT_MSGID; | |
| Parameter | Type | Description |
| Msgid | int | Message Type<br>Set MOE_DISCONNECT_MSGID |
| Result | unsigned char | Communication Result (JavaAppli Communication)<br>MOE_DISCODE_OK (0) :Normal --- Normal Disconnection<br>MOE_DISCODE_INTEROBEX (1) :Interrupted(OBEX)--- Receive DISCONNECT during WRITE or READ<br>MOE_DISCODE_INTERIR (2) :Interrupted(Ir)--- Disconnection from a lower layer<br>MOE_DISCODE_INTERUI (3) :Interrupted(UI) --- Abort by Forced Close<br>MOE_DISCODE_TIMEOUT (4) :Timeout --- Time out<br>MOE_DISCODE_ILLEGAL (5) :Illegal State --- When there is a request that cannot be processed in the not-running state<br>MOE_DISCODE_NOTACCEPTED (6) :Not Accepted --- Connection Failed<br>MOE_DISCODE_OVERFLOW (7) :Overflow --- Reception Limit Overflow<br><br>Communication Result (Infrared Native Communication)<br>MOE_DISCODE_INTERIR (2) :Interrupted(Ir)--- Disconnection from a lower layer<br>MOE_DISCODE_TIMEOUT (4) :Timeout --- Time out |
| Remark | | |
| If this message is sent, OBEX transits to the "Infrared OFF" state that is described in the State Transit Table of the "OBEX Communication Function Request Specifications". After this message is received, and after the necessary decision process is conducted, activate the AIR or retry Infrared communications or display the confirmation screen. | | |

A message that corresponds with Interrupted (OBEX)), Interrupted (Ir), Interrupted (UI), Timeout, IllegalState, Not Accepted, Overflow primitive.
<NOTE>
If the link with the lower layer is disconnected when OBEX has not received the "Client Open Request" message and the "Server Open Request" message, notify the Superior APL that the link has been disconnected using this message. The Superior APL notifies the appropriate application using this message.

Table 1-4 Communication Buffer Address Structure

| Structure Name (Type) | Structure Format | Structure Member |
|---|---|---|
| _MOE_IAPLI_BUF | typedef struct tagMOE_IAPLI_BUF{<br>unsigned char name[128];<br>unsigned char type[64];<br>unsigned char time[32];<br>unsigned char body[102400];<br>} _MOE_IAPLI_BUF; | Name Header Storage Area (name) |

Table 1-5 Header Length Setting Procedure during JavaAppli Communication

| Header | Character Code | Maximum Number of Characters or Format |
|---|---|---|
| Name(0x01) | Unicode | 63 characters or less (126 bytes) + NULL + NULL |
| Type(0x42) | ASCII | 63 characters or less (63 bytes) + NULL |
| Time(0x44) | ASCII | ISO8601 format Character string<br>YYYYMMDDTHHMMSS --- For local time (15<br>bytes)<br>YYYYMMDDTHHMMSSZ<br>--- For UTC time (16 bytes) |

- Set the number of characters (do not count the NULL character at the end) as the header length.
- If the relevant header is not received, set -1 as the header length.
- If the Time header is received in the 4 byte Time header (0xc4) format, OBEX changes it to the ISO8601 format and conducts notification.

Table 1-6 The response code that can be used during JavaAppli Communication

| Response Definition | Value | Explanation |
|---|---|---|
| Continue | 0x90 | |
| OK, Success | 0xA0 | |
| Non-Authoritative Info | 0x23 | Not used in the Server |
| Unauthorized | 0xC1 | |
| Forbidden | 0xC3 | Not used in the Server |
| Not found | 0xC4 | |
| Request Time Out | 0xC8 | Not used in the Server |
| Conflict | 0xC9 | Not used in the Server |
| Not Implemented | 0xD1 | |
| Bad Request | 0xC0 | |

| | | |
|---|---|---|
| Database Full | 0xE0 | |
| Database Locked | 0xE1 | |
| Request Entity Too Large | 0xCD | |

## 1.25 Moe_ReceiveRequest

| Function Name | Moe_ReceiveRequest | |
|---|---|---|
| Name | ReceiveRequest Function | |
| Operation Classification | JavaAppli Communication Server | |
| Functional Overview | | |
| If this function is called, OBEX will operate as if the RecieveRequest primitive is received.<br>- Wait for this function call and send a response to the DISCONNECT packet.<br>- Collect Flow point (ReceiveRequest) | | |
| Include file | moe_def.h | |
| Calling Sequence | void Moe_ReceiveRequest(void); | |
| Argument | Type | Description |
| None | | |
| Return value | Type | Description |
| None | | |
| Remark | | |
| JAM calls this function in order to notify OBEX of the timing to send the ReceiveRequest primitive. With this platform, the timing for sending a response to the DISCONNECT packet is controlled by receiving notification on the timing of the ReceiveRequest from JAVA. | | |

## 1.26 Securing process of Memory Block

| Classification | Versit Library | | |
|---|---|---|---|
| Function Name | Securing process of memory block | Symbol | Mst_VST_StartVersit |
| Functional Overview | | | |
| Secure the memory necessary for the process | | | |
| Include file | #include “VstInc.h” | | |
| Calling Sequence | unsigned short Mst_VST_StartVersit(void); | | |

| Argument | Type | I/O | Description |
|---|---|---|---|
| None | | | |

| Return value | Type | I/O | Description |
|---|---|---|---|
| | unsigned short | O | MVT_OK Normal completion<br>MVT_NO_RESRC Memory Secure Failure<br>MVT_LIB_BUSY Versit library busy |

Remark

- This function calls only once during the time between reception of the OBEC Connection Request and OBEC Disconnect or during the import/export process.
- If an abnormal completion is returned for this function, all subsequent object processes are not executable, therefore, do not call anything other than this function.
- If this function has achieved normal completion, be sure to call Mst_VST_EndVersit().

## 1.27 Release process of Memory Block

| Classification | Versit Library | | |
|---|---|---|---|
| Function Name | Release process of memory block | Symbol | Mst_VST_EndVersit |
| Functional Overview | | | |
| It is called during OBEX disconnection or import/export process completion and the secured memory is released using the Mst_VST_StartVersit function. | | | |
| Include file | #include "VstInc.h" | | |
| Calling Sequence | void Mst_VST_EndVersit(void); | | |
| Argument | Type | I/O | Description |
| None | | | |
| Return value | Type | I/O | Description |
| None | | | |
| Remark | | | |
| - This function should always be called only once during OBEX disconnection or import/export process completion. | | | |

## 1.28 Versit Library Initialization Process

| Classification | Versit Library | | |
|---|---|---|---|
| Function Name | Versit Library Initialization Process | Symbol | Mst_VST_Init |
| Functional Overview | | | |
| Conduct initialization and set the management information for each object used in the Versit Library. | | | |
| Include file | #include "VstInc.h" | | |
| Calling Sequence | unsigned short Mst_VST_Init(unsigned short obj_type, unsigned char level, unsigned char *filename, MVT_INIT_EXPRM *exprm); | | |

| Argument | Type | I/O | Description |
|---|---|---|---|
| obj_type | unsigned short | I | Object type<br>MVT_OBJ_MYPB vCard (Personal data)<br>MVT_OBJ_MYPB_ALL vCard (Personal data [show all])<br>MVT_OBJ_PB<br>MVT_OBJ_CAL vCard (Address book)<br>MVT_OBJ_EVENT vCalendar (Schedule & To Do)<br>MVT_OBJ_TODO vCalendar (Schedule only)<br>MVT_OBJ_MLD vCalendar (To Do only)<br>MVT_OBJ_IMG vNote (Melody)<br>MVT_OBJ_VCP vNote (Still image)<br>MVT_OBJ_TXT vNote (Video)<br>MVT_OBJ_INBOX vNote (Text memo)<br>MVT_OBJ_OUTBOX vMessage (Received mail)<br>MVT_OBJ_SENTBOX vMessage (Saved mail)<br>MVT_OBJ_BKM vMessage (Send mail)<br>vBookmark (Bookmark all) |
| Level | unsigned char | I | Show forwarding level<br>1: LEVEL1 forwarding<br>2: LEVEL2 forwarding |
| Filename | unsigned char * | I | Pointer of the buffer that stores the file names |
| Exprm | VST_INIT_EXPRM * | I | Pointer to the extended parameter area |

| Return value | Type | I/O | Description |
|---|---|---|---|
| | unsigned short | O | MVT_OK Normal completion<br>MVT_NG Abnormal Parameter |
| Remark | | | |

- Always call this function before sending and receiving objects and conduct initialization.

- The following processes are conducted in this function.
  (1) Clear and conduct initial setting of the Versit Condition Management Table
  (2) Set and initialize the management area of each object
- In the case of Stream type objects (SD-PIM), call Mst_VST_StreamInit() and do not call this function.

- The extended parameter (exprm) shows the following structure.

```
typedef struct tagMVT_INIT_EXPRM {
    unsigned charauthflag_g;                    /* Authentication Connection Flag      */
                                        /* ->MVT_MODE_AUTH_OFF : No Authentication */
                                        /* ->MVT_MODE_AUTH_ON : Normal Authentication Available */
                                /* ->MVT_MODE_AUTH_SYNC : IrMC-Sync Authentication
                                                        Available (ALADIN) */

    unsigned charsend_model;                    /* Send Model*/
                                        /* ->MVT_MODEL_CMN : Other than device F */
                                        /* ->MVT_MODEL_F1  : Device F */
    unsigned charDial_Data[MVT_MSISDN_DIAL_MAX+1];   /* Local Number Storage Area */
    unsigned charDev_g;                         /* Connection Device */
                                        /* ->MVT_DEV_USB : USB   */
                                        /* ->MVT_DEV_IR  : Infrared */

} MVT_INIT_EXPRM ;
```

- Concerning the File Name Storage Pointer
(1) For the File Name Storage Pointer, be sure to secure it beforehand at the call source of this function (It cannot be set at NULL pointer)
(The File Name Storage size is 128 bytes.)
(2) During LEVEL 1 import, set the file name (object or NAME header) without the extension as the file name (The path name is not necessary. Set only the file name.)Do not use the file name in cases other than LEVEL 1 import.
(3) The character codes that are allowed for file names are shown below.
ASCII, one byte katakana, Japanese characters, picture symbols
(4) The file name should end with NULL.
(5) If an internet shortcut is received, remove the last NULL and register up to 24 bytes as the title of the bookmark. (In the case of OBEX)

## 1.29 Object Reception Process

| Classification | Versit Library | | |
|---|---|---|---|
| Function Name | Object Reception Process | Symbol | Mst_VST_Write |
| Functional Overview | | | |
| Convert data that is Versit converted according to the object type and for 1 item processes, save in the temporary area and for all item processes, save in the designated storage location. | | | |
| Include file | #include "VstInc.h" | | |
| Calling Sequence | unsigned short Mst_VST_Write(unsigned char *bufp, unsigned long size,<br>unsigned short *save_type, unsigned short *num,<br>unsigned long *readsize); | | |

| Argument | Type | I/O | Description |
|---|---|---|---|
| Bufp | unsigned char * | I | OBEX Reception (Object Storage Area) Buffer |
| Size | unsigned long | I | OBEX Reception (Object Storage Area) size |
| Save_type | unsigned short * | O | Save Destination Object Type |
| Num | unsigned short * | O | Number of Registered Objects |
| readsize | unsigned long * | O | Data size read (processed) from the head of bufp |

| Return value | Type | I/O | Description |
|---|---|---|---|
| | unsigned short | O | MVT_CONTINUE Continue<br>MVT_END_OBJECT Object 1 Item Complete (SD-PIM)<br>MVT_TERMINATE All Complete<br>MVT_ERR_TERMINATE All Complete (Grammar error)<br>MVT_TEXT_ERR Versit Grammar error<br>MVT_AREA_FULL Area Full error<br>MVT_OVERLAP_ERR Overlapping error<br>MVT_NOSETTIME_ERR Mobile Device Time Not set error<br>MVT_UNKNOWN_ERR Unknown error (Fatal)<br>MVT_NO_RESRC Memory Secure Failure<br>MVT_BUFFER_OVER Versit Reception Buffer over<br>(The following is returned only in the case of OBEX.)<br>MVT_TERMINATE_RST All Complete (Reset necessary) |

| Remark | |
|---|---|

- Conduct analysis of the object data designated by the Mst_VST_Init or Mst_VST_StreamInit function and if it is
  Transfer Level 1 (1 item) then save temporarily and if it is Transfer Level 2 (All Items) then register in the Mobile device memory. The data that is temporarily saved in Transfer Level 1 (1 item) can be registered in the mobile device memory by calling the Mst_VST_DataEntry function.
- When calling this function, the Versit Library Initialization Process (Mst_VST_Init or Mst_VST_StreamInit) must
  be called. Also, if the return value is MST_VST_CONTINUE, it is necessary to recall this function but the initialization process must not be called. (The initialization process is only necessary when the object process starts.)
- The contents of the third argument is valid only when the return value of this function is MVT_TERMINATE, MVT_ERR_TERMINATE, MVT_AREA_FULL or MVT_END_OBJECT.
- The contents of the fourth argument is set as the total number of items registered of the objects processed with Versit. (When Level 2 is designated)
- The contents of the fifth argument is valid only when the return value of this function is MVT_END_OBJECT. (In the case of MVT_CONTINUE, it is processed according to the size given in the second argument and therefore it is not returned.)
- The MVT_OVERLAPPER (OVERLAP error) is reported when overlapping schedule data or URLs already exist and
  cannot be registered.
- If an overlapping schedule data or URL already exists and it cannot be registered during All item Additional import (Level10) (from SD-PIM), return MVT_OVERLAPPER and abort the import process.
- MVT_END_OBJECT returns only when it is Level 1 (1 item import) of SD-PIM.
- MVT_TERMINATE is not returned when it is Level 1 (1 item import) of SD-PIM.
- In order to show the end of import, this function must be called with "size=0" at the end. Also, if it is Level 1 (1 item import) of SD-PIM, it is the same as when the import of 1 item is complete (returned with MVT_END_OBJECT).

## 1.30 Object Send Process

| Classification | Versit Library | | |
|---|---|---|---|
| Function Name | Object Data Send Process | S y m b o l | Mst_VST_Read |
| Functional Overview | | | |
| Read the objects designated by the Mst_VST_Init function or the Mst_VST_StreamInit function from the mobile device and convert the format according to the object format. | | | |
| I n c l u d e f i l e | #include "VstInc.h" | | |
| Calling Sequence | unsigned short Mst_VST_Read(unsigned char *bufp, unsigned long size,<br>unsigned short rec_num,unsigned short folder_id,<br>unsigned short *snd_num,unsigned long *writesize); | | |
| Argument | Type | I/O | Description |
| Bufp | unsigned char * | O | OBEX Send (Object Storage Area) Buffer Address |
| Size | unsigned long | I | OBEX Send (Object Storage Area) Buffer size |
| rec_num | unsigned short | I | Record number when reading data |
| folder_id | unsigned short | I | Folder ID |
| snd_num | unsigned short * | O | Number of Send Objects |
| writesize | unsigned long * | O | Data size written (converted) from the head of bufp |
| Return value | Type | I/O | Description |
| | unsigned short | O | MVT_CONTINUE Continue<br>MVT_TERMINATE All Complete<br>MVT_OUTBOUNDS File Number Out of Bounds error<br>MVT_NOT_FOUND File Detection error<br>MVT_UNKNOWN_ERR Unknown error (Fatal)<br>MVT_APPEND_ERR there is a file attachment that cannot be sent |
| R e m a r k | | | |

- Convert the object designated by the Mst_VST_Init or Mst_VST_StreamInit function into the send format and set to
  the OBEX Send (Object Storage Area) Buffer Address.
- Convert the object format of each item in the all item process and set to the OBEX Send (Object Storage Area) Buffer Address.
- When calling this function, the Versit Library Initialization Process (Mst_VST_Init or Mst_VST_StreamInit) must
  be called first.
  Also, if the return value is MST_VST_CONTINUE, it is necessary to recall this function but the initialization process must not be called.
  (The initialization process is only necessary when the object process starts.)
- The record number designated by the third argument is only used when sending 1 item.
- At present, the folder ID designated by the fourth argument is only set in the case of video objects (in other cases, set as 0xffff).
- The total number of items that corresponds with the Progressive Bar Display function is set for the fifth argument.
- The content of the sixth argument is set as the object size written (converted) by the Versit from the head of bufp. (In the case of SD-PIM)
- MVT_APPEND_ERR shows that there is a file attachment that cannot be sent.

## 1.31 1 Item Object Registration Process

| Classification | Versit Library | | |
|---|---|---|---|
| Function Name | 1 Item Object Registration Process | Symbol | Mst_VST_DataEntry |
| Functional Overview | | | |
| Register the temporarily saved 1 item data into the memory in the mobile device. | | | |
| Include file | #include "VstInc.h" | | |
| Calling Sequence | unsigned short Mst_VST_DataEntry(unsigned short type); | | |

| Argument | Type | I/O | Description |
|---|---|---|---|
| Type | unsigned short | I | Object type<br>MVT_OBJ_PB vCard (Address book)<br>MVT_OBJ_EVENT vCalendar (Schedule only)<br>MVT_OBJ_TODO vCalendar (To Do only)<br>MVT_OBJ_MLD vNote (Melody)<br>MVT_OBJ_IMG vNote (Still image)<br>MVT_OBJ_VCP vNote (Video)<br>MVT_OBJ_TXT vNote (Text memo)<br>MVT_OBJ_INBOX vMessage (Received mail)<br>MVT_OBJ_OUTBOX vMessage (Saved mail)<br>MVT_OBJ_SENTBOX vMessage (Send mail)<br>MVT_OBJ_BKM vBookmark (Bookmark all) |

| Return value | Type | I/O | Description |
|---|---|---|---|
| | unsigned short | O | MVT_TERMINATE Normal Completion<br>MVT_ENTRY_ERR ENTRY ERROR<br>MVT_AREA_FULL Not enough area error<br>MVT_UNKNOWN_ERR Unknown error (fatal)<br>MVT_APPEND_ERR Attachment not possible |
| Remark | | | |

- Register the temporarily saved 1 item data into the memory in the mobile device.
- The argument for this function sets the contents of the third argument of the Mst_VST_Write function or the third argument of the Mst_VST_BinReceive function.
- This is reported if even one file attachment that is attached to the MVT_APPEND_ERR (Attachment Not Possible)

mail cannot be registered. If the mail cannot be registered, this code will not be reported. Also, MVT_APPEND_ERR is returned if the still image/video of the address book could not be registered because the memory was full. (Only in the case of OBEX)

## 1.32 Erase All Items Process

| Classification | Versit Library | | |
|---|---|---|---|
| Function Name | Erase All Data Process | S y m b o l | Mst_VST_EraseData |
| Functional Overview | | | |
| Erase all items of the memory data of the selected object within the mobile device. | | | |
| I n c l u d e f i l e | #include “VstInc.h” | | |
| Calling Sequence | unsigned short Mst_VST_EraseData(unsigned short obj_type); | | |

| Argument | Type | I/O | Description |
|---|---|---|---|
| obj_type | unsigned short | I | Object type<br>MVT_OBJ_PB vCard (Address book)<br>MVT_OBJ_CAL vCalendar (Schedule & To Do)<br>MVT_OBJ_EVENT vCalendar (Schedule only)<br>MVT_OBJ_TODO vCalendar (To Do only)<br>MVT_OBJ_MLD vNote (Melody)<br>MVT_OBJ_IMG vNote (Still image)<br>MVT_OBJ_VCP vNote (Video)<br>MVT_OBJ_TXT vNote (Text memo)<br>MVT_OBJ_INBOX vMessage (Received mail)<br>MVT_OBJ_OUTBOX vMessage (Saved mail)<br>MVT_OBJ_SENTBOX vMessage (Send mail)<br>MVT_OBJ_BKM vBookmark (Bookmark all) |

| Return value | Type | I/O | Description |
|---|---|---|---|
| | unsigned short | O | MVT_OK Normal completion<br>MVT_DELETE_ERR Delete Process Failure<br>MVT_NO_RESRC Memory Secure Process |
| R e m a r k | | | |

- Delete all data of the designated type (Address book schedule, To Do, original ring tone (melody), original image (still image), text memo, mail (send/receive/not sent), bookmark, original Video.

[Address book/Schedule]

- All data should be deleted regardless or the secret/non-secret mode.

[Mail]

- All data should be deleted regardless of whether or not it has been read or is protected.

[Calendar/Bookmark]

- Delete mobile device memory by calling this function during All Item import (Level 2 reception). However, in the case of vCalendar/vBookmark, (after Mst_VST_Write function is called) it is deleted when VEVENT is received (Delete TODO when VTODO is received/delete the BOX after X-NEC-BOX is received and after BOX type is determined). Therefore in the case of vCalendar/vBookmark, there are two methods of deleting all items, one using this function and one using the Mst_VST_Write function, therefore it is necessary for the Superior APL to decide which method to use to conduct All Item Delete. (If All Item Delete is conducted using this function, do not conduct All Item Delete using the Mst_VST_Write function.)

[Other]

- Delete mobile device memory by calling this function during All Item Import (Level 2 reception).

## 1.33 Send Object Size Acquisition Process

| Classification | Versit Library | | |
|---|---|---|---|
| Function Name | Get send object size process | S y m b o l | Mst_VST_Get_DbSize |
| Functional Overview | | | |
| Get the send data size of the object to be sent. | | | |
| I n c l u d e f i l e | #include "VstInc.h" | | |
| Calling Sequence | unsigned short Mst_VST_Get_DbSize(unsigned short rec_num, unsigned short folder_id, unsigned long *length, int *count); | | |

| Argument | Type | I/O | Description |
|---|---|---|---|
| rec_num | unsigned short | I | Record number when reading data |
| folder_id | unsigned short | I | Folder ID |
| length | unsigned long * | O | Total Send Byte Size |
| count | int * | O | Total Number of Send Items |

| Return value | Type | I/O | Description |
|---|---|---|---|
| | unsigned short | O | MVT_OK Normal Completion<br>MVT_NG Abnormal Completion |

R e m a r k

- Conduct abnormal completion if the object information cannot be acquired.
- When calling this function, the Versit Library Initialization Process (Mst_VST_Init or Mst_VST_StreamInit) must be called first.
- The contents of the first argument are valid only during Level 1 transfer.
- The value of the first argument is not used during Level 2 transfer. (Indeterminate Value Possible)
- Return the Total Send Byte size and Number of Send items with regards to the object data that can be sent.
- If there are no objects that can be sent, the function is considered normal complete and the byte size and number of items is 0.
- The folder ID designated in the second argument is set only when necessary (when not necessary, it is set as 0xffff).
- At present, the folder ID designated by the second argument is only set in the case of video objects (in other cases, set as 0xffff).

## 1.34 Binary Data Reception Process

| Classification | Versit Library | | |
|---|---|---|---|
| Function Name | Binary data reception process | Symbol | Mst_VST_BinReceive |
| Functional Overview | | | |
| Conduct Binary data (data other than vNote form) reception | | | |
| Include file | #include "VstInc.h" | | |
| Calling Sequence | unsigned short Mst_VST_BinReceive(unsigned char *data, unsigned long size, unsigned short *save_type); | | |

| Argument | Type | I/O | Description |
|---|---|---|---|
| Data | unsigned char * | I | OBEX Reception Buffer Pointer |
| Size | unsigned long | I | Valid Data Size |
| Save_type | unsigned short * | O | Save Destination Object Type |

| Return value | Type | I/O | Description |
|---|---|---|---|
| | unsigned short | O | MVT_CONTINUE Continue<br>MVT_TERMINATE All Complete<br>MVT_TEXT_ERR Versit Grammar Error<br>MVT_AREA_FULL Area Full Error<br>MVT_UNKNOWN_ERR Unknown error (Fatal)<br>MVT_BUFFER_OVER Versit Reception Buffer over |

Remark

- Before calling this function, call the Mst_VST_Init function and make sure initialization is conducted based on the designation of one of the objects that conduct binary transfers (objects that are not vNote format): original ringtone, original image, original video, text file, and internal line number utility.
  (If initialization is conducted under the instruction of an object that is not supported before this function is called, the operation is not guaranteed.)
- With regards to the registration of data, if it is completed with MVT_TERMINATE, it is registered by calling the Mst_VST_DataEntry function.
  However, the internal line setting utility object is Level 2 (All Item Transfer) therefore it is not necessary to call the Mst_VST_DataEntry function.
- Concerning the data format, it is determined based on the content of the binary data and stored. (Image, Video, ringtone data)

## 1.35 Object Information Acquisition Process

| Classification | Versit Library | | |
|---|---|---|---|
| Function Name | Get object information process | S y m b o l | Mst_VST_Get_ObjInfo |
| Functional Overview | | | |
| Get the object information (title) received from OBEX transfer | | | |
| I n c l u d e f i l e | #include “VstInc.h” | | |
| C a l l i n g S e q u e n c e | void Mst_VST_Get_ObjInfo(unsigned char *info); | | |
| Argument | Type | I/O | Description |
| Info | unsigned char * | O | Pointer of the object information (title) storage area |
| Return value | Type | I/O | Description |
| None | | | |

R e m a r k

- This function is valid only when the Mst_VST_Write function has normal completed in Level 1.
- Set the object information (title) that is retained in Versit during function call.
- If there is no object information, set the NULL character.
- The object information is a maximum 19 bytes, 18 characters + the NULL character.
- Be sure to secure storage area for the object information in the call source.
- The object information of SMS should be "SMS" in one byte characters.

## 1.36 Title Information Acquisition Process

| Classification | Versit Library | | |
|---|---|---|---|
| Function Name | Get title information process | S y m b o l | Mst_VST_Get_Title |
| Functional Overview | | | |
| When conducting OBEX transfer, get the title information of the object to be sent (within the mobile device memory). | | | |
| I n c l u d e f i l e | #include “VstInc.h” | | |
| Calling Sequence | unsigned short Mst_VST_Get_Title(unsigned short rec_num, unsigned short folder_id, unsigned char *title); | | |

| Argument | Type | I/O | Description |
|---|---|---|---|
| rec_num | unsigned short | I | The record number of the object that reads the title |
| folder_id | unsigned short | I | Folder ID |
| Title | unsigned char * | O | Pointer of the Title Information Storage Area |

| Return value | Type | I/O | Description |
|---|---|---|---|
| | unsigned short | O | MVT_OK Normal Completion<br>MVT_NG Abnormal Completion |

R e m a r k

- Only the title of the Bookmark can be acquired
- Be sure to secure storage area for the title information in the call source.
- If there is no title information, set the NULL character.
- The title information is a maximum 25 bytes, 24 characters + the NULL character.
- When sending 1 item of the bookmark using the Client, add an extension to the title information acquired in this function and make this the file name.
- The bookmark title outputs only shift-jis characters (with the exception of "¥", "/", ":", ";", "*", """, "<", ">", and "|").
- At present, the folder ID designated by the second argument is only set in the case of video objects (in other cases, set as 0xffff).

## 1.37 Memory Release Process

| Classification | Versit Library | | |
|---|---|---|---|
| Function Name | Memory Release Process | S y m b o l | Mst_VST_Mem_Rel |
| Functional Overview | | | |
| Release the memory secured by the mailer. | | | |
| I n c l u d e f i l e | #include “VstInc.h” | | |
| C a l l i n g S e q u e n c e | unsigned short Mst_VST_Mem_Rel(void); | | |

| Argument | Type | I/O | Description |
|---|---|---|---|
| None | | | |

| Return value | Type | I/O | Description |
|---|---|---|---|
| | unsigned short | O | MVT_OK Normal Completion<br>MVT_NG Abnormal Completion |

R e m a r k

- If a abnormality is detected in the Memory Release Process, report by Abnormal Completion.
- If there is an instruction from the user not to register in the mail registration process, this function must be called fom the Stack Control Section.
- In the case of SD-PIM, if during 1 item import at Level 1, after calling Mst_VST_Write with size=0 in the end, the Mst_VST_DataEntry function (1 item registration) is not called even though it was normal return, be sure to call this function.

(Because memory is secured using the Mst_VST_Write function and released using the Mst_VST_DataEntry function during 1 item import.)

## 1.38 Initialization Process for Each Stream Type Object

| Classification | Versit Library | | |
|---|---|---|---|
| Function Name | Versit Library Initialization Process | Symbol | Mst_VST_StreamInit |
| Functional Overview | | | |
| Conduct initialization and set the management information for each Stream type object used in the Versit Library. | | | |
| Include file | #include "VstInc.h" | | |
| Calling Sequence | unsigned short Mst_VST_StreamInit(unsigned short obj_type, unsigned char level, unsigned char *title, unsigned char *Dial_Data, MsbObject *parent); | | |

| Argument | Type | I/O | Description |
|---|---|---|---|
| obj_type | unsigned short | I | Object type<br>MVT_OBJ_PB vCard<br>MVT_OBJ_CAL vCalendar (Schedule & To Do)<br>MVT_OBJ_EVENT vCalendar (Schedule only)<br>MVT_OBJ_TODO vCalendar (To Do only)<br>MVT_OBJ_TXT vNote<br>MVT_OBJ_MESSAGE vMessage (Unknown mail)<br>MVT_OBJ_INBOX vMessage (Received mail)<br>MVT_OBJ_OUTBOX vMessage (Saved mail)<br>MVT_OBJ_SENTBOX vMessage (Send mail)<br>MVT_OBJ_BKM vBookmark<br>MVT_OBJ_MYPB Personal Data (Name, Telephone number, E-mail)<br>MVT_OBJ_MYPB_ALL Personal Data(All Data) |
| level | unsigned char | I | Shows the import /export level<br>1: 1 item import/export<br>2: Overwrite import/export of all items<br>10: All item additional import |
| filename | unsigned char * | I | Pointer of the buffer that stores the file names |
| Dial_Data | unsigned char * | I | Pointer of the buffer that stores the local number |

| Return value | Type | I/O | Description |
|---|---|---|---|
| | unsigned short | O | MVT_OK Normal Completion<br>MVT_NG Abnormal Parameter |
| Remark | | | |

- Always call this function before importing/exporting objects and conduct initialization.
- The following processes are conducted in this function.
  Clear and conduct initial setting of the Versit Condition Management Table
  Set and initialize the management area of each object
- With regards to the file name storage pointer, refer to Mst_VST_Init.

- The local number of the fourth argument should be stored at the call source when conducting export.
- MVT_OBJ_MYPB and MVT_OBJ_MYPB_ALL only supports exports. In addition, the MVT_OBJ_MYPB exports only the name,
  telephone number and mail address from the personal data and MVT_OBJ_MYPB_ALL exports all personal data.
- When calling the Mst_VST_StmGtTitle function or the Mst_VST_StreamGetObject function , designate LEVEL 1.
- The object pointer of the MSB object of the fifth argument should be set with the MSB object pointer created at the call source.
  The Versit Library creates a child object from that value as the MSB object and conducts the following MSB communication.

The difference between the all item import of level 2 and level 10

| Object | Level 2 (overwrite) | Level 10 (Additional) |
|---|---|---|
| vCard | Register the first object in the personal data. | Register the first object in the Address book. |
| vCalendar | Conduct all item delete within the Mst_VST_Write function.<br>The schedule does not check for same date setting overlapping. | Do not conduct all item delete within the Mst_VST_Write function.<br>Check for same date setting overlapping in the schedule and if overlapping data already exists in the mobile device, abort the import process. |
| vNote | - | Same as Level 2 |
| vMessage | - | Same as Level 2 |
| vBookmark | The schedule does not check for URL overlapping. | Check for URL overlapping in the schedule and if overlapping data already exists in the mobile device, abort the import process. |

- When conducting unknown vCalendar import, designate MVT_OBJ_CAL. In such cases, decide based on the BEGIN:VEVNT/BEGIN:VTODO in the object and conduct the schedule/To Do process.

- When conducting unknown vMessage import, designate MVT_OBJ_MESSAGE. In such cases, decide based on the value
  of the X-IRMC-BOX property within the object and conduct receive/send/save mail processes. (If there is no X-IRMC-BOX property or the value is invalid, treat it as a received mail.

- If either one of "MVT_OBJ_INBOX, MVT_OBJ_OUTBOX, or MVT_OBJ_SENTBOX" is designated, destroy the X-IRMC-BOX
  property within the object and conduct the process in accordance with the designated object type.

## 1.39 Stream Type Object Title Acquisition Process

| Classification | Versit Library | | |
|---|---|---|---|
| Function Name | Get title information process | S y m b o l | Mst_VST_StreamGetTitle |
| Functional Overview | | | |
| Conduct Stream type object title information acquisition | | | |
| I n c l u d e f i l e | #include "VstInc.h" | | |
| Calling Sequence | unsigned short Mst_VST_StreamGetTitle(unsigned char *bufp, unsigned long size,<br>unsigned char *title, unsigned long *offset,<br>unsigned long *objectsize, unsigned short obj_type); | | |

| Argument | Type | I/O | Description |
|---|---|---|---|
| Bufp | unsigned char * | I | Object Data Storage Address |
| Size | unsigned long | I | Object Data Size |
| Title | unsigned char * | O | Title Storage Area Address (32 bytes) |
| offset | unsigned long * | O | The offset value of the front position of the discovered object from bufp |
| objectsize | unsigned long * | O | The size of the discovered object |
| obj_type | unsigned short | O | Discovered Object Type<br>MVT_OBJ_PB vCard<br>MVT_OBJ_EVENT vCalendar (Schedule only)<br>MVT_OBJ_TODO vCalendar (To Do)<br>MVT_OBJ_TXT vNote<br>MVT_OBJ_INBOX Received mail(E-mail)<br>MVT_OBJ_OUTBOX Saved mail (E-mail)<br>MVT_OBJ_SENTBOX Sent mail(E-mail)<br>MVT_OBJ_INBOX_SMS Received mail(SMS)<br>MVT_OBJ_OUTBOX_SMS Saved mail (SMS)<br>MVT_OBJ_SENTBOX_SMS Sent mail(SMS)<br>MVT_OBJ_BKM vBookmark |

| Return value | Type | I/O | Description |
|---|---|---|---|
| | unsigned short | O | MVT_OK Normal Completion<br>MVT_CONTINUE Continue<br>MVT_NG Abnormal Completion (such as object error, EOF detection) |
| R e m a r k | | | |

- It also accommodates SD-PIM standard data.
- If the object is not found, set 0xffffffff in the fourth argument (offset).
- If the object data ends midway or the object data starts midway, the data size processed to that point is stored in the fifth argument (objectsize). (If the data is divided, the objectsize returned during the first call and the objectsize returned during the second call combined make up the actual object size.)
- The title storage area should be secured in the call source.
- The information to be set in the title storage area for each object is shown below.

(1) vCard

The last name and first name of the N value should be shown connected (with no space in between). (If there is no N, set as FN.)

(2) vCalendar

Show the DESCRIPTION value.

(3) vNote

Show the contents of the text memo.

(4) vMessage

In the case of SMS, show the content of the text and in the case of E-mail, show the Subject.

(5) vBookmark

Show the Bookmark title.

- A maximum of 31 bytes with a NULL character (¥0) as the last data is stored for the title.
- If there is no title, a NULL character (¥0) is stored as the first character of the Title Storage Area.
- If the object data is divided, return MVT_CONTINUE.
- This function must be called in the end using "size=0" in order to show the end of the object file.

## 1.40 Stream Type Object Detailed Information acquisition Process

| Classification | Versit Library | | |
|---|---|---|---|
| Function Name | Get object information process | Symbol | Mst_VST_StreamGetObject |
| Functional Overview | | | |
| Conduct Stream type object detailed information acquisition | | | |
| Include file | #include "VstInc.h" | | |
| Calling Sequence | unsigned short Mst_VST_StreamGetObject(unsigned char *bufp , unsigned longsize , unsigned short obj_type, void *data); | | |

| Argument | Type | I/O | Description |
|---|---|---|---|
| Bufp | unsigned char * | I | Object Data Storage Top Address |
| Size | unsigned long | I | Object Data Size |
| obj_type | unsigned short | I | Object type<br>MVT_OBJ_PB vCard<br>MVT_OBJ_EVENT vCalendar (Schedule only)<br>MVT_OBJ_TODO vCalendar (To Do)<br>MVT_OBJ_TXT vNote<br>MVT_OBJ_INBOX Received mail(E-mail)<br>MVT_OBJ_OUTBOX Saved mail (E-mail)<br>MVT_OBJ_SENTBOX Sent mail(E-mail)<br>MVT_OBJ_INBOX_SMS Received mail(SMS)<br>MVT_OBJ_OUTBOX_SMS Saved mail (SMS)<br>MVT_OBJ_SENTBOX_SMS Sent mail(SMS)<br>MVT_OBJ_BKM vBookmark |
| Data | void * | O | Pointer of the Object Information Storage Area |

| Return value | Type | I/O | Description |
|---|---|---|---|
| | unsigned short | O | MVT_OK Normal Completion<br>MVT_CONTINUE Continue<br>MVT_NG Abnormal Completion (Object Error) |

| Remark |
|---|

- Always store the offset value acquired from the Mst_VST_StreamGetTitle in the bufp and the readsize in the size
  and conduct call.
- For the third argument, always set and call the object type (the sixth argument) acquired in Mst_VST_StreamGetTitle.
- The correspondence between the object type (the third argument) and the Information Storage Area Structure (the fourth argument) is shown below.

(1) vCard(MVT_OBJ_PB)
    MVT_VCARD_DATA *
(2) vCalendar(MVT_OBJ_EVENT)
    MVT_VEVENT_DATA *
(3) vCalendar(MVT_OBJ_TODO)
    MVT_VTODO_DATA *
(4) vNote
    MVT_VNOTE_DATA *
(5) vMessage(MVT_OBJ_INBOX, MVT_OBJ_OUTBOX, MVT_OBJ_SENTBOX)
    MVT_VMESSAGE_DATA *
(6) vMessage(MVT_OBJ_INBOX_SMS, MVT_OBJ_OUTBOX_SMS, MVT_OBJ_SENTBOX_SMS)
    MVT_SMS_DATA *
(7) vBookmark(MVT_OBJ_BKM)
    MVT_VBOOKMARK_DATA *

- The area of the structures above should be secured in the call source.
(-> For details, refer to Appendix A. Structure.)
- If after this function returns "MVT_CONTINUE", the end of the object file (EOF) is detected, this function must be called in the end using "size=0".

# 2. MAW

Details of MAW function referred externally from other tasks are described below.

## 2.1 Get End Message Display Status (Maw_EndImage_ref)

| Classification | MAW function | | No. | 1 |
|---|---|---|---|---|
| Function name | Get end message display status | | Symbol | Maw_EndImage_ref |
| Calling side | E-lib (Application manager) | | | |
| Outline of functions | | | | |
| Refers to goodbye message display prohibition flag | | | | |
| Include file | maw_p_stadef.h | | | |
| Calling sequence | WORD Maw_EndImage_ref(void); | | | |
| Argument name | Type | I/O | Description | |
| — | — | | — | |
| Return value | Type | I/O | Description | |
| sMaw_NO_ByeBye_Msg_FLG | WORD | | Flag that prohibits the display of goodbye message hold by MAW | |
| Subordinate function | Type | | | |
| | | | | |
| Remark | | | | |

## 2.2 Equipment Power Status Output (Maw_Offcharge_Status)

| Classification | MAW function | | No. | 2 |
|---|---|---|---|---|
| Function name | Equipment power status output | | Symbol | Maw_Offcharge_Status |
| Calling side | E-lib (packet communication, voice communication, equipment) | | | |
| Outline of functions | Output to another AP if the MAW status is equipment power ON or OFF | | | |
| Include file | maw_p_stadef.h | | | |
| Calling sequence | SHORT Maw_Offcharge_Status(void); | | | |
| Argument name | Type | I/O | Description | |
| — | — | — | — | |
| Return value | Type | I/O | Description | |
| sts | SHORT | O | Equipment power status<br>ON : Equipment power ON<br>OFF : Equipment power OFF | |
| Subordinate function | Type | | | |
| | | | | |
| Remark | * 1 When the charging status is equipment power ON (MAW_CHARGE_POWON), returns ON (equipment power ON).<br><br>* Equipment power: Equipment power ON status is the power ON status in which user can operate the keys.On the other hand, equipment power OFF status is the status that user sees the power is OFF while the power is being supplied. | | | |

## 2.3 User Agent Readout (Maw_Useragent)

| Classification | MAW function | | No. | 4 |
|---|---|---|---|---|
| Function name | User agent readout | | Symbol | Maw_Useragent |
| Calling side | JAM, Mailer | | | |
| Outline of functions | Copy user agent to the specified area | | | |
| Include file | maw_p_stadef.h | | | |
| Calling sequence | VOID Maw_Useragent( Data_Area ); | | | |
| Argument name | Type | I/O | Description | |
| Data_Area | BYTE * | O | User agent data storage area | |
| Return value | Type | I/O | Description | |
| — | — | — | — | |
| Subordinate function | Type | | | |
| | | | | |
| Remark | | | | |

## 2.4 Power OFF Sequence Status Readout Processing (Maw_Powoff_StsRef)

| Classification | MAW function | | No. | 5 |
|---|---|---|---|---|
| Function name | Powr OFF sequence status readout processing | | Symbol | Maw_Powoff_StsRef |
| Calling side | E-lib | | | |
| Outline of functions | | | | |
| Processing that notifies in which sequence the freeze occurs after the power OFF freeze monitoring timer time-out | | | | |
| Include file | maw_p_stadef.h | | | |
| Calling sequence | BYTE Maw_Powoff_StsRef( void ); | | | |
| Argument name | Type | I/O | Description | |
| — | — | — | — | |
| Return value | Type | I/O | Description | |
| MAW_POWOFFSTS_COM<br>MAW_POWOFFSTS_APM<br>MAW_POWOFFSTS_SRV<br>MAW_POWOFFSTS_SVC | BYTE<br>BYTE<br>BYTE<br>BYTE | O<br>O<br>O<br>O | Power is ON<br>Wait for communication end confirmation phase 1<br>Wait for communication end confirmation phase 2<br>Wait for communication end confirmation phase 3 | |
| Subordinate function | Type | | | |
| | | | | |
| Remark | | | | |

## 2.5 Get Handset Starting Mode (Maw_StartMode_Status)

| Classification | MAW function | | No. | 8 |
|---|---|---|---|---|
| Function name | Get handset starting mode | | Symbol | Maw_StartMode_Status |
| Calling side | elib(lmp).monster(term).taf(src).wm(csrc) | | | |
| Outline of functions | Maw_StartMode_Status() returns handset starting mode. | | | |
| Include file | maw_p_prim.h 、maw_p_stadef.h | | | |
| Calling sequence | int Maw_StartMode_Status(void) | | | |
| Argument name | Type | I/O | Description | |
| — | — | — | — | |
| Return value | Type | I/O | Description | |
| MMI_NORMAL<br>MMI_TEST<br>MMI_SPECIAL | int<br>int<br>int | O<br>O<br>O<br>O<br>O<br>O<br>O | Normal mode<br>Test mode<br>Special mode | |
| Subordinate function | Type | | | |
| | | | | |
| Remark | | | | |
| | | | | |

## 2.6 Backlight Status Setting (Maw_BackLightLevel_set)

| Classification | MAW function | | No. | 9 |
|---|---|---|---|---|
| Function name | Backlight on status setting | | Symbol | Maw_BackLightLevel_set |
| Calling side | elib(lmp).elib(wdc) | | | |
| Outline of functions | Maw_BackLightLevel_set() switches the battery threshold table according to the backlight status specified by vol. | | | |
| Include file | maw_p_stadef.h | | | |
| Calling sequence | int Maw_BackLightLevel_set(int vol) | | | |
| Argument name | Type | I/O | Description | |
| vol | int | I | Backlight status<br>Set the following values<br>MAW_BLT_FOFF light off<br>MAW_BLT_OFF slight light<br>MAW_BLT_LOW level 1 (low)<br>MAW_BLT_MID level 2 (middle)<br>MAW_BLT_HGH level 3 (high) | |
| Return value | Type | I/O | Description | |
| MAW_API_OK<br>MAW_API_NG<br>MAW_API_PERR1 | int<br>int<br>int | O<br>O<br>O | Normal<br>Internal error (system call error, etc)<br>Argument error (incorrect vol ) | |
| Subordinate function | Type | | | |
| | | | | |
| Remark | When the backlight is turned on/off, the status should be set by this API in real time. | | | |

## 2.7 Camera Power Status Setting (Maw_CamPower_set)

| Classification | MAW function | | No. | 10 |
|---|---|---|---|---|
| Function name | Camera power status setting | | Symbol | Maw_CamPower_set |
| Calling side | 3G-324M（VH）、mm_plugin(VideoCapture) | | | |
| Outline of functions | Maw_CamPower_set() switches the battery threshold table according to the camera status specified by vol. | | | |
| Include file | maw_p_stadef.h | | | |
| Calling sequence | int Maw_CamPower_set(int pow) | | | |
| Argument name | Type | I/O | Description | |
| pow | int | I | Camera power status setting<br>The following values should be set<br>MAW_CAM_POW_ON camera power ON<br>MAW_CAM_POW_DOWN camera power OFF | |
| Return value | Type | I/O | Description | |
| MAW_API_OK<br>MAW_API_NG<br>MAW_API_PERR1 | int | O | Normal<br>Internal error (system call error, etc)<br>Argument error (incorrect pow) | |
| Subordinate function | Type | | | |
| | | | | |
| Remark | When the camera is started/stopped, the status should be set by this API in real time. | | | |